Subject: FAQ: CSH Coke Machine Information

This article was archived around: Sat, 4 Feb 1995 19:45:09 GMT

All FAQs in Directory: Root Directory
All FAQs posted in: alt.folklore.college, alt.folklore.computers, alt.internet.services
Source: Usenet Version

```
Archive-name: csh-coke-machine-info
Version: $Id: FAQ,v 1.2 1994/05/23 15:57:24 pat Exp pat $
Posting-Frequency: monthly
```

Contents:

1. _A_Brief_Overview_of_The_Computer_Science_House_Coke_Machine_.

The following was only slightly modified from a mail message written by Tad Hunt (Wed, 27 Apr 1994 12:25:54 -0400)

Our drink system is composed of several parts, the drink machine itself, the computer inside the machine, the serial connection to our drink server machine, the accounting software, and finally the wide area information systems (such as finger).

The computer inside the drink machine is very simple. It accepts commands to "drop" drinks from slots 1-5 (by sending an ascii '1'-'5' from the server), and a command to query status (an ascii 'A'). The computer drops drinks by tripping a relay to the solenoid on the particular slot requested. The status returned is a bit pattern indicating if each slot is either full or empty (full is defined as one or more cans remaining) based on sensors in the slots. All of this can be done through the wires that normally interface to the buttons on the front of the machine.

On the server machine, there is a program called "drink" which keeps track of money in user accounts, how many cans are in each slot, the kind of drink in each slot, and how long the drink has been in the machine (for determining how cold it is). This is the only program with permission to access "/dev/drink", which is the serial port the Coke Machine is connected to. If the serial port isn't locked, it locks it, then querys the status of the machine to display a menu to the user, and allows them to choose a drink. If the user has enough money in their account, and the slot isn't marked empty, the program asks for a time delay to wait before dropping the drink (in case the user needs to walk to the

machine), then sends the command down the serial line to drop the appropriate drink.

Finally comes the wide area information systems. We setup an alias for one of our machines (satan.csh.rit.edu) to answer requests for drink.csh.rit.edu, and rewrote the finger program to display machine status by rsh(1)ing to the server machine and running the drink program in a mode to query machine statistics. Also added at this point was an XWindows drink client program, which uses command line options to "drink" to allow users to run an XWindows drink machine program and drop drinks by clicking a button. You can run this program yourself -- if you have XWindows -- by doing the following: "finger $DISPLAY@drink.csh.rit.edu", it will send "xdrink" to your display. Also, you can get a graph of machine statistics by doing: "finger graph@drink.csh.rit.edu"

The drink system will soon be entering the "client-server" age with the addition of a debit daemon written by one of the drink support people, which will be useful for more than just drink, it's a generic debit system. Most if not all of this system will soon be available for anonymous ftp from ftp.csh.rit.edu in pub/drink.

2. _The_History_Of_Our_Machine_.

Somebody here at RIT threw away a Coke Machine. It was pretty beat up, but the members of Computer Science House plucked it from the trash none-the-less. The Coke Machine was cleaned up and put to use. It ran as a normal vending machine for some time in this way.

But, as the red-tape flies, the company who owns the vending machine rights to the RIT campus complained that we were threatening their rights. In a wonderous swirl of politics and crazy techies resulted in the Coke Machine being hooked up to the computer systems. For, as you see, a 'Vending Machine' is a machine that accepts money and gives out consumables in return. We don't have a 'Vending Machine' so much as a high-tech group refrigerator. The Coke Machine only accepts money or returns a drink. If you're silly enough to put money in the little slot, you've lost your money. If you've already given money to a drink admin, you can dispense a drink through the computer systems.

Not long after that, a newer Coke Machine was donated to Computer Science House. The first implementations of the Coke Machine were done on a small processor on a bread-board. The newer implementation is a bit more 'rugged' (and explained below) in that many of the connections are actually soldered. 8^).

This machine has been painted the CSH colors (purple and pink a la DEC/pdp). It bears the Computer Science House name. We're proud to have it on the Internet, but we must admit that CMU beat us to the punch. Our big advantage over their machine though is the ability to drop a drink from where you sit and have it arrive at the same time you get to the machine. Our machine has been the subject of little blurbs in major publications across the country and is listed as the 7th-most-fingered site by Wired magazine. And, the current record for long-distance drops is Arizona to Rochester.

3. _The_Hardware_In_The_Machine_.

The computer in the machine is a small 8051 board with a serial connection, LCD display, A/D convertor, and several out ports. The EPROM that this board runs from contains code written by Sean McGranaghan. That code is loosely based on code written by Frank

Giuffrida for that board's intended purpose as power-supply monitor and regulator.

As is mentioned above, this software simply reads the status lines that were at one time hooked up to the LED indicators on the buttons of the machine to check the fullness of a slot. In its current incarnation, the sensor on the Jolt slot tends to stick in the 'Empty' position. Fortunately for us caffeine mongers, the 'Empty' indicator can be ignored.

If a request for status is received on the serial line, a bit mask is formed indicating which slots are full. This bit mask is sent back over the serial line to the waiting program that made the request. In this mask, the bits 0 through 4 are used to represent the 5 slots on the machine. The 5th bit is also set to ensure that the return value is a printable ascii character and as a verification that it actually did check the slots.

If an ascii digit on the range '1' - '5' is sent to this board, it triggers a solid-state relay which closes the circuit that would normally be closed by pressing the button on the front of the machine. If this is successful, a 'D' is transmitted back to the waiting program. If this fails, an 'E' is sent back to the waiting program. ('D' is for drop. 'E' is for error.)

The LCD on the board constantly displays a message 'CSH Coke Machine' and the amount of time since the board has been reset in the form 'Day 000 00:00:00'.

Currently, several schemes are being considered for this board to verify that it is talking to some program and not to someone with tip(1) access to the device. This will make the Coke Machine no more vulnerable to root attack than user attack.

4. _The_Software_On_the_Unix_Side_.

The board in the Coke Machine is connected to a CCI Power 6/32 Tahoe that is currently running BSD 4.3 Shanzer (a custom blend of 4.3 Reno, 4.3 Tahoe, and 4.4 Alpha). /dev/drink is configured as a 9600-baud connection to the board in the Coke Machine. The connection is over a standard RS232 connection.

The software consists, currently, of one main program called 'drink'. drink(1) maintains a database of user balances and statistics as well as slot statistics. This software has undergone many revisions (read: total rewrites) over the years. I'm pretty sure that I'm the only one who will admit to having touched it. But, I'll drag in Bob Krzaczek's name into it to as the last person to touch the stuff I hacked on.

In its current incarnation, this software keeps track of user balances in CSH-franks. These bear a striking relation to US-dollars in that the exchange rate has always been 100 CSH-franks to 1 U.S. Dollar. But, such relationships are human constructions and probably just coincidences that reflect deep underlying symmetries in the web of the Universe (or not) [much like the way RIT student ids resemble, but are distinct from, social security numbers]. CSH-franks are known by some as CSHmids and CSH-wonder-wubbas. But... the last 6 minutes of voting turned up 2 votes for CSH-franks, 1.5 votes for CSH-bobas, several incoherent mumbles about Pink Floyd, a compromise for CSH-verypinkmetaloidoncebelongedtofrankthenBOBAwonderwubbamids, and not much else in the close-to-relevant category. [If Ross Perot was ever part of a CSH wall(1) war, he'd think twice about

electronic town-meetings.]

The drink(1) program offers several command line options. These are:

-o [12345rg]	where a number specifies a slot to drop a drink from, 'r' specifies to drop a drink from a random slot (choosing from the full-ones), and 'g' is a special gamble option (to be described later).
-d N	delays for N seconds before dropping the drink.
-l login	useful for dropping a drink from the balance of the user given by 'login'. This option prompts you for a password to validate you.
-m	forces menu mode where the current slot statistics are displayed.
-b	shows the user's balance.
-s	shows the user's raw statistics as number of drinks dropped per slot.
-S	shows the user's statistics in relation to the global statistics. This options shows number of drinks dropped by the user on a per-slot basis and the number of drinks dropped overall on a per-slot basis and the percentage per-slot the user makes up. For example, for me, now, I have dropped 382 of 2412 drinks dropped from slot 5 since last time the statistics were cleared. I account for 15.8375 percent of the drops from slot 5 (Coke Classic). Also given in these statistics is the current gamble cost and the accumulated gamble (to be explained later).
-t	shows the number of drinks in each slot divided into time slots. Along the vertical, each slot is shown. Across the horizontal, the number of drinks in the slot for less than one hour, between one and three hours, more than three hours, and the total in the slot is shown.
-T	This option is similar to the last but it puts out the information in a form easily readable by other programs. First, it puts out the current time as returned by time(2). Then, it puts out on the following the title of the first slot and a string representing how many drinks are in the slot and what times they were placed there. This string is of the form: number time number time number time 0 All time(2) values are printed in hexidecimal. These couplets are repeated for each slot.

The gamble option was originally designed to make use of the fact that the drink machine isn't always full and the fact that people may not have enough of a balance to afford a drink. With the gamble option, the cost of gambling is computed by adding up the prices of all of the full slots, dividing by the total number of slots and adding the 'gamble cost'. The 'gamble cost' is currently 2 CSH-franks. This is a fudge-factor to favor drink staying in the black. The more slots that are empty, the lower the cost. The current risk is 12 CSH-franks, but the odds of getting a drink are only 1 in 5

(and that slot is the Diet Mystery Slot (a double whammee)). A side goal is to integrate a 'Coke' as a potion in some deep dungeon in nethack and, if you quaff it, and the accumulated gamble cost is greater than than the price of a Coke, it'll drop you one. The current accumulated gamble cost is 262 CSH-franks.

5. _The_Wide-Area_Software_.

If you have finger(1) access, X access, or Mosaic(1) access, you can witness our Coke Machine first-hand from where you sit. There are two main pathways to our Coke Machine through the Internet. These are through Tad Hunt's modified finger(1) @drink.csh.rit.edu and through Eric Van Hensbergen's xdrink(1) interface. Both of these can be accessed from the CSH Drink Machine page on the World Wide Web. The URL for that page is:

http://www.csh.rit.edu/proj/drink.html

5a) graph

One of the first bits of net access we allowed to our Coke Machine was to finger(1) it to get information. The current state of the Coke Machine can be divined in several ways through finger(1). Tad Hunt rewrote the finger(1) program at drink.csh.rit.edu to handle several virtual users. The first of these can be accessed by fingering graph@drink.csh.rit.edu. This will display an informational message and an ascii representation of the Coke Machine. This representation includes the price at each slot, the number of drinks in each slot, and a graph representing the coldness of those drinks. Emprical tests have shown that complete coldness of a drink is achieved in three hours.

A different view of the contents of the coke machine can be obtained by fingering drink@drink.csh.rit.edu. There is a great deal of redundant information in this, but.... what can you do?

Fingering info@drink.csh.rit.edu will provide more information into the ways to finger the coke machine.

And, if you're running X-windows, a command like 'finger mymachine.mynet.myorg:0.0@drink.csh.rit.edu' or 'finger ${DISPLAY}@drink.csh.rit.edu' should bring up an X-interface to our drink software on your display.

b) xdrink

The xdrink(1) interface was written by Eric Van Hensbergen. It presents bitmaps for each of the slots (they easily get out of date (sorry)). It offers a pointy-clicky interface for those not too keen on command lines. It represents the fullness of each slot with a bar-graph, the contents with a bitmap, and the mystery-slot as a flashing pattern of the bitmaps.

6. _Current_Plans_for_the_Drink_Machine_.

I am currently almost finished with a new incarnation of the drink software. This incarnation involves a 'telnet' interface similar to those of 'smtp' and 'ftp'. It also provides a means for kerberos authentication. Additionally, it will talk several languages from English to German to Esperanto to Lojban to Rot13 (English) to Piglatin (English). More on this as it's available. Also, in this incarnation, one will be able to risk any amount of money above the gamble cost, choosing their desired slot, and having odds proportional to the amount risked divided by the cost of the desired drink.

For more further questions or to arrange a personal tour of Computer Science House, mail drink@drink.csh.rit.edu.
--
"Hot Hands of an Oslo Dentist"